

Assignment 4-2 Polynomial Multiplication 2

You are given two polynomials $f(x)$ and $g(x)$ with integer coefficients. In this problem you'll have to find out the product $f(x) \cdot g(x)$ of $f(x)$ and $g(x)$. For the sake of convenience, we define the zero polynomial as $0x^0$.

Input

The input consists of t ($30 \leq t \leq 40$) test cases. The first line of the input contains only positive integer t . Then t test cases follow. Each test case consists of two polynomials $f(x)$ and $g(x)$. Each polynomial consists of three lines: the first line contains only one positive integer m ($m \leq 10$) which represents the number of nonzero terms of the polynomial; the second line consists of m nonzero integers (in the range $[-2^{25}, 2^{25} - 1]$) representing the coefficients of the polynomial; the third line consists of m integers (in the range $[0, 2^{62} - 1]$ and in decreasing order) representing the corresponding exponents of the polynomial. For example, the polynomial $300x^{1000} + 200x^{100} + 100x^{10}$ is represented as the following three lines:

```
3
300 200 100
1000 100 10
```

You may assume that neither $f(x)$ nor $g(x)$ is the zero polynomial.

Output

For each test case, you are to output three lines representing $f(x) \cdot g(x)$, in the same format as the input.

Sample Input

```
2
3
1 -2 -4
3 2 0
1
4
0
3
-1 -2 4
3 2 0
4
1 2 1 -7
```

3 2 1 0

Sample Output

2
1 -2
3 2
2
1 -3
1 0

Requirements

In your program, a polynomial $a_n x^{d_n} + a_{n-1} x^{d_{n-1}} + \dots + a_2 x^{d_2} + a_1 x^{d_1} + a_0 x^{d_0}$ should be represented by the following two arrays:

coefficient array:

n	$n-1$	\dots	2	1	0
a_n	a_{n-1}	\dots	a_2	a_1	a_0

exponent array:

n	$n-1$	\dots	2	1	0
d_n	d_{n-1}	\dots	d_2	d_1	d_0

For example, the polynomial $-x^3 - 2x^2 + 4$ is represented by the following two arrays:

coefficient array:

2	1	0
-1	-2	4

exponent array:

2	1	0
3	2	0

Note that $d_n > d_{n-1} > \dots > d_2 > d_1 > d_0 \geq 0$, and for every $i = 0, 1, \dots, n$, $a_i \neq 0$.

Part of the program

Please write the function addition and multiplication to complete the following program which solves this problem. In your program, you cannot declare global variables (including global arrays) except arraySize. Moreover, you cannot declare arrays (or vectors) in the function addition, and you can declare at most one array (or vector) in the function multiplication.

```
// Polynomial multiplication
#include <iostream>
using namespace std;

// product = multiplicand * multiplier
void multiplication( long long int multiplicandCoef[], long long int multiplicandExpon[],
                    int multiplicandSize,
                    long long int multiplierCoef[], long long int multiplierExpon[],
```

```

        int multipliersSize,
        long long int productCoef[], long long int productExpon[],
        int &productSize );

// addend += adder
void addition( long long int addendCoef[], long long int addendExpon[], int &addendSize,
              long long int adderCoef[], long long int adderExpon[], int adderSize );

// returns true if and only if the specified polynomial is the zero polynomial
bool isZero( long long int coefficient[], int size );

// outputs the specified polynomial
void output( long long int coefficient[], long long int exponent[], int size );

const int arraySize = 10;

int main()
{
    int T;
    cin >> T;
    for( int t = 0; t < T; t++ )
    {
        long long int multiplicandCoef[ arraySize ] = {};
        long long int multiplicandExpon[ arraySize ] = {};
        int multiplicandSize;
        cin >> multiplicandSize; // input multiplicand
        for( int i = multiplicandSize - 1; i >= 0; i-- )
            cin >> multiplicandCoef[ i ];
        for( int i = multiplicandSize - 1; i >= 0; i-- )
            cin >> multiplicandExpon[ i ];

        long long int multiplierCoef[ arraySize ] = {};
        long long int multiplierExpon[ arraySize ] = {};
        int multipliersSize;
        cin >> multipliersSize; // input multiplier
        for( int i = multipliersSize - 1; i >= 0; i-- )
            cin >> multiplierCoef[ i ];
        for( int i = multipliersSize - 1; i >= 0; i-- )
            cin >> multiplierExpon[ i ];

        long long int productCoef[ arraySize * arraySize ] = {};
        long long int productExpon[ arraySize * arraySize ] = {};
        int productSize = 1;

        // product = multiplicand * multiplier
        multiplication( multiplicandCoef, multiplicandExpon, multiplicandSize,
                      multiplierCoef, multiplierExpon, multipliersSize,
                      productCoef, productExpon, productSize );

        output( productCoef, productExpon, productSize );
    }
}

// product = multiplicand * multiplier
void multiplication( long long int multiplicandCoef[], long long int multiplicandExpon[],
                   int multiplicandSize,
                   long long int multiplierCoef[], long long int multiplierExpon[],
                   int multipliersSize,
                   long long int productCoef[], long long int productExpon[],
                   int &productSize )
{

}

// addend += adder
void addition( long long int addendCoef[], long long int addendExpon[], int &addendSize,
              long long int adderCoef[], long long int adderExpon[], int adderSize )
{

}

// returns true if and only if the specified polynomial is the zero polynomial
bool isZero( long long int coefficient[], int size )

```

```

{   return ( size == 1 && coefficient[ 0 ] == 0 );
}

// outputs the specified polynomial
void output( long long int coefficient[], long long int exponent[], int size )
{
    cout << size << endl;
    cout << coefficient[ size - 1 ];
    for( int i = size - 2; i >= 0; i-- )
        cout << " " << coefficient[ i ];
    cout << endl;

    cout << exponent[ size - 1 ];
    for( int i = size - 2; i >= 0; i-- )
        cout << " " << exponent[ i ];
    cout << endl;
}

```